SYSTEM AND METHOD FOR MANIPULATING OFFLINE SOFTWARE

TECHNICAL FIELD

The present invention relates to the field of software manipulation. In particular, this invention relates to a system and method for manipulation by a computer of offline target software stored on one or more target computer-readable media of a target computer.

BACKGROUND OF THE INVENTION

In some prior art systems, installing and configuring desirable features of a software product is complex and requires a significant length of time. In such systems, customizing an operating system and its installed applications can only be done by running that operating system and performing actions while in that context. Such online system manipulation actions include installing or uninstalling applications, customizing settings such as desktop wallpaper, or adding a new user. Booting into the operating system to make these changes is a time consuming process and requires validation of those changes thereby delaying deployment of the operating system. In addition, updating files or system settings for software that is actively executing may result in the software becoming inconsistent or unstable. Further, reboots may be required between updates.

In addition, the prior art systems fail to support scenarios in which an operating system requires certain changes prior to booting. For example, the operating system may require a certain device driver. In such cases, the prior art systems perform the changes after the operating system boots.

For these reasons, a system for offline system manipulation is desired to address one or more of these and other disadvantages.

SUMMARY OF THE INVENTION

The invention provides software for manipulation by a computer of offline target software stored on one or more target computer-readable media of a target computer. In particular, the invention includes a driver executing on the computer to provide access to the offline target software.

The invention includes software functionality that allows installation and configuration of an operating system and/or associated application programs offline or otherwise outside the context of the operating system associated with the target software. That is, the invention includes a software installation and configuration technique performed without installing or traditionally running the target software or an operating system associated with the target software.

With the architecture of the invention, installation and configuration actions, data store manipulation, execution of application programming interface routines, file system manipulation, and other actions can be applied to an operating system and application programs while the operating system and application programs are in an offline state. Software of the invention redirects such actions performed in an online manner to the location of the files associated with the offline system. In addition, if a list of the actions is declared or otherwise available to the invention software, the invention software provides for performance of the actions natively (i.e., without redirection). The architecture of the invention can be used in both an online and offline manner to act on the operating system and application programs of the running system or on a set of files at another location, respectively.

With the invention, the user can modify software on one or more target computers without booting into an operating system associated with the software. In addition, changes can be made to the software without re-validating the software in its entirety or otherwise re-packaging the software.

In accordance with one aspect of the invention, a system provides manipulation of target software by a computer. The target software is stored on one or more target computer-readable media. The target software has an online state and an offline state. The system executes a driver on the computer to provide access to the target software. The driver includes one or more redirect components for manipulating the target software when the target software is offline. The manipulation occurs in response to at least one command received from a user.

In accordance with another aspect of the invention, a method provides manipulation of target software by a computer. The target software is stored on one or more target computer-readable media. The target software has an online state and an offline state. The method further includes accessing the target computer-readable media when the target software is offline in response to at least one command received from a user.

In accordance with another aspect of the invention, a computer-readable medium includes a data structure. The data structure represents a queue and is for use by a computer in manipulating target software stored on one or more target computer-readable media. The target software has an online state and an offline state. The manipulation occurs when the target software is in the offline state and in response to at least one command received from a user. Further, the data structure includes a queue field that stores a list of one or more tasks associated with the executable command.

In accordance with yet another aspect of the invention, a system adds one or more drivers to one or more target computer-readable media. The system performs the addition via a computer. The target software has an online state and an offline state. The system includes a user interface associated with the computer and adapted for creating a system preparation file. The system preparation file includes a list of the drivers to be added to the target software responsive to user input. The system further includes a software tool that executes on the computer. The software tool is responsive to the user interface and installs the drivers listed in the system preparation file to the target software when the target software is in the offline state.

Alternatively, the invention may comprise various other methods and apparatuses.

Other features will be in part apparent and in part pointed out hereinafter.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is an exemplary block diagram illustrating a computer connected to a target computer-readable media.

FIG. **2** is an exemplary block diagram illustrating a target computer-readable medium storing target software.

FIG. **3** is an exemplary block diagram illustrating the driver software components of the computer communicating with target software.

FIG. **4** is an exemplary flow chart illustrating operation of the driver software components.

FIG. **5** is a block diagram illustrating an exemplary computer-readable medium storing a queue.